PROJ.4 - General Parameters

This document attempts to describe a variety of the PROJ.4 parameters which can be applied to all, or many coordinate system definitions. This document does not attempt to describe the parameters particular to particular projection types. Some of these can be found in the GeoTIFF Projections Transform List. The definitative documentation for most parameters is Gerald's original documentation available from the main PROJ.4 page.

False Easting/Northing

Virtually all coordinate systems allow for the presence of a false easting (+x_0) and northing (+y_0). Note that these values are always expressed in meters even if the coordinate system is some other units. Some coordinate systems (such as UTM) have implicit false easting and northing values.

---

pm - Prime Meridian

A prime meridian may be declared indicating the offset between the prime meridian of the declared coordinate system and that of greenwich. A prime meridian is clared using the "pm" parameter, and may be assigned a symbolic name, or the longitude of the alternative prime meridian relative to greenwich.

Currently prime meridian declarations are only utilized by the pj_transform() API call, not the pj_inv() and pj_fwd() calls. Consequently the user utility **cs2cs** does honour prime meridians but the **proj** user utility ignores them.

The following predeclared prime meridian names are supported. These can be listed using the cs2cs argument **-lm**.

```
   greenwich 0dE
      lisbon 9d07'54.862"W
       paris 2d20'14.025"E
      bogota 74d04'51.3"E
      madrid 3d41'16.48"W
        rome 12d27'8.4"E
        bern 7d26'22.5"E
     jakarta 106d48'27.79"E
       ferro 17d40'W
    brussels 4d22'4.71"E
   stockholm 18d3'29.8"E
      athens 23d42'58.815"E
        oslo 10d43'22.5"E
```

Example of use. The location long=0, lat=0 in the greenwich based lat/long coordinates is translated to lat/long coordinates with Madrid as the prime meridian.

```
 cs2cs +proj=latlong +datum=WGS84 +to +proj=latlong +datum=WGS84 +pm=madrid
0 0                           (input)
3d41'16.48"E    0dN 0.000     (output)
```

---

towgs84 - Datum transformation to WGS84

Datum shifts can be approximated by 3 parameter spatial translations (in geocentric space), or 7 parameter shifts (translation + rotation + scaling). The parameters to describe this can be described using the **towgs84** parameter.

In the three parameter case, the three arguments are the translations to the geocentric location in meters.

For instance, the following demonstrates converting from the Greek GGRS87 datum to WGS84.

```
% cs2cs +proj=latlong +ellps=GRS80 +towgs84=-199.87,74.79,246.62 \
    +to +proj=latlong +datum=WGS84
20 35
20d0'5.467"E    35d0'9.575"N 8.570
```

The EPSG database provides this example for transforming from WGS72 to WGS84 using an approximated 7 parameter transformation.

```
% cs2cs +proj=latlong +ellps=WGS72 +towgs84=0,0,4.5,0,0,0.554,0.219 \
    +to +proj=latlong +datum=WGS84
4 55
4d0'0.554"E     55d0'0.09"N 3.223
```

The seven parameter case uses *delta_x*, *delta_y*, *delta_z*, *Rx - rotation X*, *Ry - rotation Y*, *Rz - rotation Z*, *M_BF - Scaling*. The three translation parameters are in meters as in the three parameter case. The rotational parameters are in seconds of arc. The scaling is apparently the scale change in parts per million.

A more complete discussion of the 3 and 7 parameter transformations can be found in the EPSG database (trf_method's 9603 and 9606). Within PROJ.4 the following calculations are used to apply the **towgs84** transformation (going to WGS84). The x, y and z coordinates are in geocentric coordinates. Three parameter transformation (simple offsets):

```
  x[io] = x[io] + defn->datum_params[0];
  y[io] = y[io] + defn->datum_params[1];
  z[io] = z[io] + defn->datum_params[2];
```

Seven parameter transformation (translation, rotation and scaling):

```
  #define Dx_BF (defn->datum_params[0])
  #define Dy_BF (defn->datum_params[1])
  #define Dz_BF (defn->datum_params[2])
  #define Rx_BF (defn->datum_params[3])
  #define Ry_BF (defn->datum_params[4])
  #define Rz_BF (defn->datum_params[5])
  #define M_BF  (defn->datum_params[6])

  x_out = M_BF*(       x[io] - Rz_BF*y[io] + Ry_BF*z[io]) + Dx_BF;
  y_out = M_BF*( Rz_BF*x[io] +       y[io] - Rx_BF*z[io]) + Dy_BF;
  z_out = M_BF*(-Ry_BF*x[io] + Rx_BF*y[io] +       z[io]) + Dz_BF;
```

Note that EPSG method 9607 (coordinate frame rotation) coefficients can be converted to EPSG method 9606 (position vector 7-parameter) supported by PROJ.4 by reversing the sign of the rotation vectors. The methods are otherwise the same.

---

nadgrids - Grid Based Datum Adjustments

In many places (notably North America and Austrialia) national geodetic organizations provide grid shift files for converting between different datums, such as NAD27 to NAD83. These grid shift files include a shift to be applied at each grid location. Actually grid shifts are normally computed based on an interpolation between the containing four grid points.

PROJ.4 currently supports use of grid shift files for shifting between datums and WGS84 under some circumstances. The grid shift table formats are ctable (the binary format produced by the PROJ.4 nad2bin program), NTv1 (the old Canadian format), and NTv2 (.gsb - the new Canadian and Australian format).

Use of grid shifts is specified using the "nadgrids" keyword in a coordinate system definition. For example:

```
% cs2cs +proj=latlong +ellps=clrk66 +nadgrids=ntv1_can.dat \
    +to +proj=latlong +ellps=GRS80 +datum=NAD83 << EOF
-111 50
EOF
111d0'2.952"W   50d0'0.111"N 0.000
```

In this case the /usr/local/share/proj/ntv1_can.dat grid shift file was loaded, and used to get a grid shift value for the selected point.

It is possible to list multiple grid shift files, in which case each will be tried in turn till one is found that contains the point being transformed.

```
% cs2cs +proj=latlong +ellps=clrk66 \
          +nadgrids=conus,alaska,hawaii,stgeorge,stlrnc,stpaul \
    +to +proj=latlong +ellps=GRS80 +datum=NAD83 << EOF
-111 44
```

```
EOF
111d0'2.788"W   43d59'59.725"N 0.000
```

Skipping Missing Grids

The special prefix **@** may be prefixed to a grid to make it optional. If it not found, the search will continue to the next grid. Normally any grid not found will cause an error. For instance, the following would use the ntv2_0.gsb file if available, otherwise it would fallback to using the ntv1_can.dat file.

```
% cs2cs +proj=latlong +ellps=clrk66 +nadgrids=@ntv2_0.gsb,ntv1_can.dat \
    +to +proj=latlong +ellps=GRS80 +datum=NAD83 << EOF
-111 50
EOF
111d0'3.006"W   50d0'0.103"N 0.000
```

The null Grid

A special **null** grid shift file is shift with releases after 4.4.6 (not inclusive). This file provides a zero shift for the whole world. It may be listed at the end of a nadgrids file list if you want a zero shift to be applied to points outside the valid region of all the other grids. Normally if no grid is found that contains the point to be transformed an error will occur.

```
% cs2cs +proj=latlong +ellps=clrk66 +nadgrids=conus,null \
    +to +proj=latlong +ellps=GRS80 +datum=NAD83 << EOF
-111 45
EOF
111d0'3.006"W   50d0'0.103"N 0.000
```

```
% cs2cs +proj=latlong +ellps=clrk66 +nadgrids=conus,null \
    +to +proj=latlong +ellps=GRS80 +datum=NAD83 << EOF
-111 44
-111 55
EOF
111d0'2.788"W   43d59'59.725"N 0.000
111dW   55dN 0.000
```

Downloading and Installing Grids

The source distribution of PROJ.4 contains only the ntv1_can.dat file. To get the set of US grid shift files it is necessary to download an additional distribution of files from the PROJ.4 site, such as proj-nad27-1.1.tar.gz. Overlay it on the PROJ.4 source distribution, and re-configure, compile and install. The distributed ASCII .lla files are converted into binary (platform specific) files that are installed. On windows using the `nmake /f makefile.vc nadshift` command in the `proj\src` directory to build and install these files.

It appears we can't redistribute the Canadian NTv2 grid shift file freely, though it is better than the NTv1 file. However, end users can download it for free from the NRCan web site at http://www.geod.nrcan.gc.ca/software/ntv2_e.php. After downloading it, just dump it in the data directory with the other installed data files (usually /usr/local/share/proj).

Caveats

1. Where grids overlap (such as conus and ntv1_can.dat for instance) the first found for a point will be used regardless of whether it is appropriate or not. So, for instance, +nadgrids=ntv1_can.dat,conus would result in the canadian data being used for some areas in the northern United States even though the conus data is the approved data to use for the area. Careful selection of files and file order is necessary. In some cases border spanning datasets may need to be pre-segmented into Canadian and American points so they can be properly grid shifted.

2. There are additional grids for shifting between NAD83 and various HPGN versions of the NAD83 datum. Use of these haven't been tried recently so you may encounter problems. The FL.lla, WO.lla, MD.lla, TN.lla and WI.lla are examples of high precision grid shifts. Take care!

3. Additional detail on the grid shift being applied can be found by setting the PROJ_DEBUG environment variable to a value. This will result in output to stderr on what grid is used to shift points, the bounds of the various grids loaded and so forth.

4. PROJ.4 always assumes that grids contain a shift **to** NAD83 (essentially WGS84). Other types of grids might or might not be usable.